This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Cancelled)

2. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the executable file is compiled by a compiler from a C-type programming language object model document.

3. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the loader upon instantiating the security agent provides same with a second reference to the instantiated object model, whereby the commander does not have the second reference and therefore cannot directly access the object model or command same to act.

4. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the instantiated security agent passes on each command from the commander to the object model unless such security agent deems such command to be of a type that should not be so passed on.

5. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the security agent does not pass on to the object model a type of command that would expose the object model in a non-obfuscated form.

6. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the security agent does not pass on to the object model a type of command that would expose the object model with a level of granularity finer than a pre-defined maximum.

7. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the security agent passes on to the object model a substitute command that

exposes the object model with a level of granularity coarser than the pre-defined maximum.

8. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the loader instantiates the security agent separately from the object model.

9. (Currently Amended) ~~The document of claim 1~~ The system of claim 34, wherein the loader instantiates the security agent as part of the object model.

10. (Currently Amended) A method for loading a persisted object model from an object model document, the method comprising:

providing a compiled executable file having an image source, a security source, and a loader;

instantiating the loader in a memory of a computer upon a command from a commander to execute the executable file to instantiate the persisted object model;

the loader instantiating the object model in the memory from the image source;

the loader instantiating a security agent in the memory from the security source, the security agent ~~for controlling~~ limiting access to the object model as instantiated in the memory of the computer, wherein the security agent does not allow the object model to be exposed in a non-obfuscated form; and

the loader returning to the commander a first reference to the instantiated security agent, whereby the commander in employing the first reference accesses the security agent rather than the instantiated object model.

11. (Original) The method of claim 10 further comprising the loader upon instantiating the security agent providing same with a second reference to the instantiated object model, whereby the commander does not have the second reference

and therefore cannot directly access the object model or command same to act.

12. (Original) The method of claim 10 further comprising the instantiated security agent passing on each command from the commander to the object model unless such security agent deems such command to be of a type that should not be so passed on.

13. (Original) The method of claim 12 comprising the security agent not passing on to the object model a type of command that would expose the object model in a non-obfuscated form.

14. (Original) The method of claim 12 comprising the security agent not passing on to the object model a type of command that would expose the object model with a level of granularity finer than a pre-defined maximum.

15. (Original) The method of claim 14 comprising the security agent passing on to the object model a substitute command that exposes the object model with a level of granularity coarser than the pre-defined maximum.

16. (Original) The method of claim 10 comprising the loader instantiating the security agent as part of the object model.

17. (Original) The method of claim 10 comprising the loader instantiating the security agent as part of the object model.

18. (Currently Amended) A computer-readable storage medium having stored thereon instructions, which when executed, instantiate a loader for an object model document for persisting an object model therein and enable a commander to indirectly access the object model, the object model document comprising a compiled executable file having an executable image source file, ~~a~~ an executable security source, and ~~a~~ an executable loader, the instructions comprising:

instantiating a loader in memory of a computer using the executable loader;

instantiating ~~the image source from which~~ the persisted object model ~~is instantiated~~ in ~~a~~ the memory of ~~a~~ the computer using the executable image source;

instantiating ~~the security source from which~~ a security agent ~~is instantiated~~ in the memory of the computer using the executable security source, wherein [[;]] the security agent ~~for controlling~~ limits access to the object model as instantiated in the memory of the computer such that the security agent does not allow the object model to be exposed in a non-obfuscated form; and ~~the loader, which instantiates the executable file in the memory of the computer upon a command from a commander, the loader instantiating the object model in the memory from the image source, the loader instantiating the security agent in the memory from the security source,~~ wherein the instructions return to the commander a first reference to the instantiated security agent, whereby the commander in employing the first reference accesses the security agent rather than the instantiated object model.

19. (Original) The medium of claim 18 wherein the executable file is compiled by a compiler from a C-type programming language object model document.

20. (Original) The medium of claim 18 wherein the loader upon instantiating the security agent provides same with a second reference to the instantiated object model, whereby the commander does not have the second reference and therefore cannot directly access the object model or command same to act.

21. (Original) The medium of claim 18 wherein the instantiated security agent passes on each command from the commander to the object model unless such security such security agent deems such command to be of a type that should not be so passed on.

22. (Original) The medium of claim 21 wherein the security agent does not pass on to the object model a type of command that would expose the object model in a non-obfuscated form.

23. (Original) The medium of claim 21 wherein the security agent does not pass on to the object model a type of command that would expose the object model with a level of granularity finer than a pre-defined maximum.

24. (Original) The medium of claim 23 wherein the security agent passes on to the object model a substitute command that exposes the object model with a level of granularity coarser than the pre-defined maximum.

25. (Original) The medium of claim 18 wherein the loader instantiates the security agent separately from the object model.

26. (Original) The medium of claim 18 wherein the loader instantiates the security agent as part of the object model.

27-33. (Cancelled)

34. (New) A computer system comprising:

an input device that receives a command input from a user to display information from an object model via a software application having an application commander;

a compiled executable file having an executable image file source, an executable security source, and an executable loader,

wherein upon execution of the executable file, (a) an object model is instantiated in a memory of the system using the executable image source, (b) a security agent is instantiated in the memory of the computer using the security source, the security agent controlling access to the object model as instantiated in the memory of the computer, and (c) the loader is instantiated in the memory of the computer using the executable loader, and wherein during execution of the executable file, a first reference is generated for the instantiated security agent to point to the object model and a second reference is generated for the application commander to point to the instantiated security agent, and wherein, in response to an application request for information from the object model:

(i) the application commander employs the first reference to access the security agent rather than the object model, and

(ii) the security agent limits access to the object model before accessing the requested information from the object model using the second reference; and

a computer monitor that displays the requested information, wherein the requested information from the object model is provided by the security agent to the application commander if the request for information does not act to expose the object model in a non-obfuscated form.